On the Continuity of Cascaded Generalized Inverse Redundancy Resolution, with Application to Kinematically Redundant Manipulators

Travis Baratcart, Valerio Salvucci, and Takafumi Koseki
Department of Electrical Engineering and Information Systems, The University of Tokyo, Japan
b_travis@koseki.t.u-tokyo.ac.jp

***Abstract*—Redundancy in systems is valued for its ability to add additional dexterity in task completion, but its utilization implies an added degree of complexity which must be resolved at runtime. The majority of resolution methods have focused on resolution using some variant of the 2-norm optimizing pseudo-inverse. While tractable, the pseudo-inverse fails to consider input bounds and often resolves systems with input values unrealizable given physical parameters. The Cascaded Generalized Inverse (CGI) was introduced to incorporate maximum input bounds and extend the output space of the pseudo-inverse by iteratively reallocating joint contributions to underutilized inputs. This paper discusses the discontinuities which arise in resolution of systems utilizing CGI, and proposes a constrained CGI, which overcomes this discontinuity. Proof of continuity is provided, and the constrained and unconstrained CGI systems are simulated and compared with application to kinematically-redundant robotic manipulators.**

I. Introduction

Redundant systems are those which possess a greater number of degrees of freedom than are required to fulfill the task. These additional degrees of freedom allow for declaration of subtasks to be fulfilled in addition to the primary task associated with the system. While redundancy is something to be desired in terms of dexterity, it also introduces a complication in translation from desired output to the input space which must be resolved, raising questions of continuity and utility of the solution.

The problem of redundancy is considered as follows: The matrix $\mathbf{B} \in \Re^{mxn}$ translates actuation variables $\mathbf{u} \in \Re^n$ to task space values $\mathbf{v} \in \Re^m$ as follows:

$$\mathbf{v} = \mathbf{B}\mathbf{u} \tag{1}$$

In the case that $m = n$ and $\mathbf{B}$ is non-singular, the problem of acquiring input values from a desired output is simple: $\mathbf{B}$ is invertible and yields a unique solution. However, in a redundant system ($n > m$) an infinite number of solutions exist for any particular output. This problem of selecting a unique, continuous solution with some utility amongst the infinitely available options has been the subject of a great deal of research.

The vast majority of solutions have focused on resolution using some variant of the Moore-Penrose psuedo-inverse, which resolves (1) as

$$\mathbf{u} = \mathbf{B}^{\dagger}\mathbf{v} \tag{2}$$

where if $\mathbf{B}$ is full row rank, the pseudo inverse, $\mathbf{B}^{\dagger}$, of $\mathbf{B}$ is defined as follows:

$$\mathbf{B}^{\dagger} := \mathbf{B}^T(\mathbf{B}\mathbf{B}^T)^{-1} \tag{3}$$

The psuedo-inverse approach returns a unique continuous solution for $\mathbf{u}$, resulting in the 2-norm optimization of the inputs, subject to (1), or

$$\min(\sqrt{u_1^2 + u_2^2 + \ldots + u_n^2}) \tag{4}$$

From a physical perspective, as a minimization of the sum of squares, a parallel can be drawn between the solution attained and minimization of energy in the system. This relationship is strengthened through consideration of the weighted pseudo-inverse [1], formed by weighting the pseudo-inverse by some criteria matrix, as

$$\bar{\mathbf{B}}^{\dagger} = \mathbf{W}^{-1}\mathbf{B}^T(\mathbf{B}\mathbf{W}^{-1}\mathbf{B}^T)^{-1} \tag{5}$$

However, the pseudo-inverse is far more often utilized for its analytical tractability than for its physical properties. As such, there are situations when utilizing psuedo-inverse redundancy resolution yields values which may not be ideal in consideration of the physical conditions. One of these cases is when the individual actuator contributions need to be limited. As the pseudo-inverse is employed by minimizing a sort of average of inputs, it does not explicitly concern itself with the inputs on an individual level. This can often lead to a single actuator shouldering a disproportionately large share of the total output. If that value exceeds the maximum input contribution, then the corresponding output exists outside the solution space for 2-norm, and the system will fail to complete the specified task. This forces one to restrict the system to only a subset of those physically attainable outputs.

Many researchers have put forth alternative resolution methods to solve this problem, such as optimization utilizing the infinity-norm [2], [3], neural networks [4], [5], and nonlinear phase-different control [6], [7]. These methods offer such benefits as full output utilization and increased speed. But ultimately the pseudo-inverse's dexterity and ease of

978-1-4799-0224-8/13/$31.00 ©2013 IEEE

implementation has led to the domination of pseudo-inverse based methods in the resolution of redundancy.

When system constraints render pseudo-inverse resolution unideal, rather than changing resolution schemes, system designers typically make use of methods to extend the range of the pseudo-inverse.

This paper will review these methods of extending the output range of the pseudo-inverse, and in particular will discuss the Cascaded Generalized Inverse (CGI), which offers the greatest increase in output range. It is found, however, that in attempting to extend the output range of the pseudo-inverse, the Cascaded Generalized Inverse loses one of the main benefits of pseudo-inverse resolution, continuity. A domain restriction will be proposed ensuring continuity, and comparison of the two systems conducted.

II. Methods of Extending the Output Range of the Pseudo-Inverse

A. Least-Squares with Clipping

Effectively ignoring the upper limits, least-squares with clipping [8] simply truncates any input values in excess of their bounds. Least-squares with clipping is a particularly undesirable method of extending the output space of the pseudo-inverse as it will fail to produce the desired trajectory to be resolved.

B. The Redistributed Pseudo-Inverse

The Redistributed Pseudo-Inverse [9] successfully extends the range of the Pseudo-Inverse (for a period) without resolution error. The method is described as follows:

1) Find the pseudo-inverse resolution of the system. If all resolved inputs lie within their output bounds, the process ends.
2) If any resolved inputs lie outside their output bounds, the corresponding inputs are truncated to the respective maximums.
3) The new system created, corresponding to the unmaximized inputs and the desired output minus the contributions of the maximized inputs, is then resolved using the pseudo-inverse.

The Redistributed Pseudo-Inverse is a significant step up from the pseudo-inverse, though it falls victim to the same issues of least-squares with clipping in the second level of optimization.

C. The Cascaded Generalized Inverse

Given the failing in the second level of optimization of the Redistributed Pseudo-Inverse, it seems reasonable then to repeat the process on the second level of optimization and repeat until the result is feasible. The described process is referred to as the Cascaded Generalized Inverse [10]. The Cascaded Generalized Inverse iteratively continues resolving the system until the objective function is satisfied or all inputs have been saturated. Variations exist where the process continues until the resultant system is represented by an invertible square matrix, one variable is arbitrarily saturated in each iteration in the interest of speed [11], and the ”most” saturated variable is saturated each iteration to preserve directionality at the expense of speed [12].

The Cascaded Generalized Inverse has and is still being applied to many problems including aircraft control allocations [13], [14], VTOL control systems [15], ship berthing [16], and ship positioning systems [17]. But, absent in the literature is a discussion on the discontinuities arising in resolution when utilizing CGI resolution. Depending on the application, discontinuities in input resolution could have large impacts on the governed system. For example, in one of the prior discussed CGI applications, the ship berthing system [16], part of the problem statement includes the fact that the tugboats utilized are slow to change their applied force and bearing. Should the discontinuities associated with CGI resolution arise in the control distribution scheme of this system, it would be very slow to respond leading to large error.

To avoid such complications, this paper seeks to call attention to the issue of discontinuity in CGI resolution and pose an addendum to the method preventing discontinuity for those systems which cannot tolerate discontinuity in their control structure.

III. Discontinuities in Cascaded Generalized Inverse Resolution

To illustrate this problem, we will simulate such a case in which discontinuity in resolution arises. The considered system is the four-link kinematically-redundant planar manipulator.

Kinematic-redundancy is valued in robotic applications for its ability to add additional dexterity to task completion, allowing for such behaviors as obstacle [18] and singularity avoidance [19].

The problem of kinematic-redundancy is formulated as follows. The Jacobian, $\mathbf{J}$, relates joint velocities, $\dot{\mathbf{q}}$, to end-effector velocity, $\mathbf{v}_{eff}$ as follows:

$$\mathbf{v}_{eff} = \mathbf{J}\dot{\mathbf{q}} \tag{6}$$

where

$$[\mathbf{J}]_{(i,j)} = \frac{\delta[\mathbf{x}_{eff}]_i}{\delta q_j} \tag{7}$$

So, in the case of the four-link planar manipulator, there exist four joint space variables to actuate the 2-dimensional end-effector velocity.

For this simulation, and simulations to follow, define configuration A as the following:

Configuration A :=

$L_1 = L_2 = L_3 = L_4 = 1$ (unit)

$\mathbf{q} = [\frac{\pi}{32}, \frac{\pi}{6}, \frac{\pi}{6}, \frac{\pi}{6}]$ (rad)

$\dot{\mathbf{q}}_{max} = [1, 2, 10, 10]$ (rad/sec)

A manipulator in configuration A, as shown in Fig. 1, is prompted for increasing end-effector velocity in the direction $\theta = 0$. Fig. 2 illustrates the results of redundancy resolution

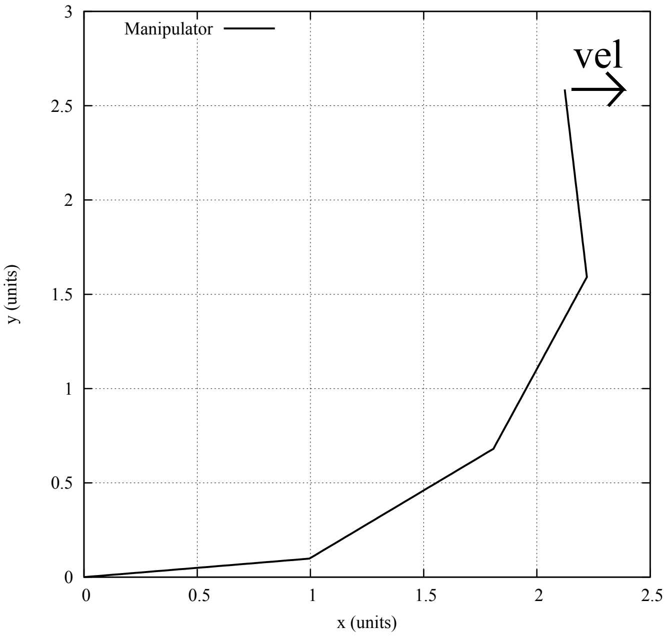

Fig. 1. Manipulator in configuration A, used for demonstration of discontinuity in CGI Resolution

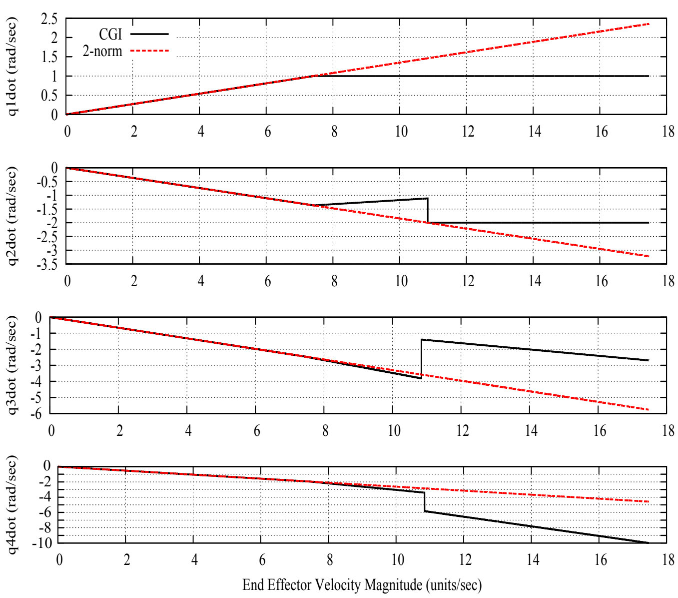

Fig. 2. CGI resolved joint velocities for increased end-effector velocity magnitude in configuration A

carried out by the 2-norm optimizing pseudo-inverse and the cascading pseudo-inverse method.

It is seen that when the 2-norm resolution of $\dot{q}_2$ exceeds the maximum limit, at $\|\mathbf{v}\| \approx 11$ units/sec, the discrepancy between this resolution and second-level resolution associated with maximized $\dot{q}_1$ causes a discontinuity affecting joints q_2, q_3, and q_4. However, discontinuous velocity changes are impossible to realize in a physical system. Attempting to realize them anyway will lead to such negative effects as joint knocking [20], [21] and deviation from the intended path— temporary so long as an appropriate controller is utilized.

Such negative effects will manifest to varying degree in all systems when discontinuities surface in their control system, so when possible it is beneficial to design our systems so that they remain continuous over their operating regions. As such, we will propose a restriction on the domain of the cascaded pseudo-inverse

IV. Proposal of Constrained CGI to Ensure Continuity

It is noted that discontinuity occurs when there is a disconnect between the resolution of an input when it exceeds its limits, and the resolution achieved while re-optimized when other variables exceed their limits. It seems reasonable then, that if only one input is permitted to exceed its limit, that these discontinuities might be avoided.

The above may be implemented for any LTI mxn system as follows:

Let $\mathbf{B} \in \Re^{m,n}$ relate input values $\mathbf{u} \in \Re^{n}$ to task space values $\mathbf{v} \in \Re^{m}$ as in (1).

Define $\mathbf{B}_i$ as the i'th column of $\mathbf{B}$, $\mathbf{B}'_{i_1.i_2...i_z}$ as the matrix $\mathbf{B}$ with the i_1, i_2, ... , and i_z'th column removed. Likewise define u_i as the i'th element of $\mathbf{u}$, and $\mathbf{u}'_{i_1.i_2...i_z}$ as the vector $\mathbf{u}$ with the i_1, i_2, ... , and i_z'th element removed.

Parentheticals will be used following input values to denote the resolution of the inputs in the corresponding case. For example, $u_3(1(\text{a}))$ and $u'_3(1(\text{a}))$ refer respectively to the values of the third element of $\mathbf{u}$ and the vector $\mathbf{u}$ with the third element removed attained in the case 1(a). If inputs are referred to without parentheticals, it will refer to the value attained from the currently discussed case.

For given $\mathbf{v}$ select the elements of $\mathbf{u}$ as:

Case 0 :

$$\mathbf{u} = \mathbf{B}^{\dagger}\mathbf{v} \tag{8}$$

Case $i_1(\text{a})$:

$$u_{i_1} = u_{i_1 max} \tag{9}$$

$$\mathbf{u}'_{i_1} = (\mathbf{B}'_{i_1})^{\dagger}(\mathbf{v} - \mathbf{B}_{i_1} u_{i_1 max}) \tag{10}$$

Case $i_1(\text{b})$

$$u_{i_1} = -u_{i_1 max} \tag{11}$$

$$\mathbf{u}'_{i_1} = (\mathbf{B}'_{i_1})^{\dagger}(\mathbf{v} + \mathbf{B}_{i_1} u_{i_1 max}) \tag{12}$$

Case $i_1(x_1).i_2(x_2)...i_{z-1}(x_{z-1}).i_z(\text{a})$:

$$u_{i_1} = \begin{cases} u_{imax} & \text{if } x_1 = \text{a} \\ -u_{imax} & \text{if } x_1 = \text{b} \end{cases} \tag{13}$$

$$\vdots$$

$$u_{i_{z-1}} = \begin{cases} u_{i_{z-1} max} & \text{if } x_{z-1} = \text{a} \\ -u_{i_{z-1} max} & \text{if } x_{z-1} = \text{b} \end{cases} \tag{14}$$

$$u_{i_z} = u_{i_z max} \tag{15}$$

$$\mathbf{u}'_{i_1...i_z} = (\mathbf{B}'_{i_1...i_z})^{\dagger}(\mathbf{v} - \sum_{\gamma=1}^{z} \mathbf{B}_{i_\gamma} u_{i_\gamma}) \tag{16}$$

Case $i_1(x_1).i_2(x_2)...i_{z-1}(x_{z-1}).i_z(\mathrm{b})$:

$$u_{i_1} = \begin{cases} u_{i_1 max} & \text{if } x_1 = \mathrm{a} \\ -u_{i_1 max} & \text{if } x_1 = \mathrm{b} \end{cases} \tag{17}$$

$$\vdots$$

$$u_{i_{z-1}} = \begin{cases} u_{i_{z-1} max} & \text{if } x_{z-1} = \mathrm{a} \\ -u_{i_{z-1} max} & \text{if } x_{z-1} = \mathrm{b} \end{cases} \tag{18}$$

$$u_{i_z} = -u_{i_z} \tag{19}$$

$$\mathbf{u}'_{i_1...i_z} = (\mathbf{B}'_{i_1...i_z})^\dagger (\mathbf{v} - \sum_{\gamma=1}^{z} \mathbf{B}_{i_\gamma} u_{i_\gamma}) \tag{20}$$

and define

$$\text{case } 0 := |u_i(0)| \leq u_{imax}, \forall i \in [1, n] \tag{21}$$

$$\text{case i(a)} := u_i(0) > u_{imax}, |u_j(0)| \leq u_{jmax}, \forall j \neq i \tag{22}$$

$$\text{case i(b)} := u_i(0) < -u_{imax}, |u_j(0)| \leq u_{jmax}, \forall j \neq i \tag{23}$$

$$\begin{aligned} &\text{Case } i_1(x_1).i_2(x_2)...i_{z-1}(x_{z-1}).i_z(\mathrm{a}) := \\ &\quad (\text{Case } i_1(x_1).i_2(x_2)...i_{z-1}(x_{z-1})), \\ &\quad (u_{i_z}(i_1(x_1).i_2(z_2)...i_{z-1}(x_{z-1})) > u_{i_z max}), \\ &\quad \text{and } (|u_{i_j}(i_1(x_1).i_2(z_2)...i_{z-1}(x_{z-1}))| \leq u_{i_j max}, \forall j \neq z) \end{aligned} \tag{24}$$

$$\begin{aligned} &\text{Case } i_1(x_1).i_2(x_2)...i_{z-1}(x_{z-1}).i_z(\mathrm{b}) := \\ &\quad (\text{Case } i_1(x_1).i_2(x_2)...i_{z-1}(x_{z-1})), \\ &\quad (u_{i_z}(i_1(x_1).i_2(x_2)...i_{z-1}(x_{z-1})) < -u_{i_z max}), \\ &\quad \text{and } (|u_{i_j}(i_1(x_1).i_2(x_2)...i_{z-1}(x_{z-1}))| \leq u_{i_j max}, \forall j \neq z) \end{aligned} \tag{25}$$

We will consider cases with more pseudo-inverse evaluation iterations to be of higher order than those with less. If a system satisfies class conditions for multiple classes across multiple levels, then we say the system is operating in the highest order class which it satisfies.

V. Proof of Continuity of Constrained CGI

Here we will demonstrate continuity of the constrained Cascaded Generalized Inverse.

Proof: Let $\mathbf{B}$ be an mxn, full row rank matrix which translates actuation variables $\mathbf{u} \in \Re^n$ to task space values $\mathbf{v} \in \Re^m$ as in (1). Let $\mathbf{B}_i$, $\mathbf{B}'_{i_1.i_2...i_z}$, u_i, and $\mathbf{u}'_{i_1.i_2...i_z}$ be as originally defined in section IV. We will first demonstrate continuity within each individual case and then demonstrate continuity during switching of cases, proving full piece-wise continuity.

In case 0 (when all 2-norm satisfying inputs are within their respective maximum bounds), the solution is equivalent to the pseudo-inverse solution. If $\mathbf{B}$ is full row rank, the solution is continuous within case 0 by the continuity property of the pseudo-inverse.

In case $i_1(x_1).i_2(x_2)...i_{z-1}(x_{z-1}).i_z(\mathrm{a})$, the solution for $\mathbf{u}$ is of the form:

$$u_{i_1} = \begin{cases} u_{i_1 max} & \text{if } x_1 = \mathrm{a} \\ -u_{i_1 max} & \text{if } x_1 = \mathrm{b} \end{cases} \tag{26}$$

$$\vdots$$

$$u_{i_{z-1}} = \begin{cases} u_{i_{z-1} max} & \text{if } x_{z-1} = \mathrm{a} \\ -u_{i_{z-1} max} & \text{if } x_{z-1} = \mathrm{b} \end{cases} \tag{27}$$

$$u_{i_z} = u_{i_z max} \tag{28}$$

$$\mathbf{u}'_{i_1.i_2...i_z} = (\mathbf{B}'_{i_1...i_z})^\dagger (\mathbf{v} - \sum_{\gamma=1}^{z} \mathbf{B}_{i_\gamma} u_{i_\gamma}) \tag{29}$$

In this region, $u_{i_1}...u_{i_z}$ are constants which are clearly continuous, and $\mathbf{u}'_{i_1.i_2...i_z}$ may be rewritten as

$$\mathbf{u}'_{i_1.i_2...i_z} = (\mathbf{B}'_{i_1...i_z})^\dagger \mathbf{v} - (\mathbf{B}'_{i_1...i_z})^\dagger \sum_{\gamma=1}^{z} \mathbf{B}_{i_\gamma} u_{i_\gamma}) = (\mathbf{B}'_{i_1...i_z})^\dagger \mathbf{v} + \mathbf{k} \tag{30}$$

where $\mathbf{k}$ is a constant vector. If $\mathbf{B}'_{i_1...i_z}$ is full rank, $\mathbf{u}'_i$ is linear with respect to $\mathbf{v}$, by the continuity property of the pseudoinverse, and is therefore continuous.

A similar method shows continuity throughout case $i_1(x_1).i_2(x_2)...i_{z-1}(x_{z-1}).i_z(\mathrm{b})$.

Now that continuity has been demonstrated within each individual case, we now turn to demonstrating switching continuity.

While in case $i_1(x_1).i_2(x_2)...i_{z-1}(x_{z-1})$, the system is prompted to enter case $i_1(x_1).i_2(x_2)...i_{z-1}(x_{z-1}).i_z(\mathrm{a})$ when the value resolved in case $i_1(x_1).i_2(x_2)...i_{z-1}(x_{z-1})$ for input u_{i_z} becomes greater than $u_{i_z max}$. We must therefore demonstrate continuity at the condition:

$$\begin{aligned} &(\text{Case } i_1(x_1).i_2(x_2)...i_{z-1}(x_{z-1})), \\ &(u_{i_z}(i_1(x_1).i_2(x_2)...i_{z-1}(x_{z-1})) = u_{i_z max}), \\ &\text{and } (|u_{i_j}(i_1(x_1).i_2(x_2)...i_{z-1}(x_{z-1}))| \leq u_{i_j max}, \forall j \neq i) \end{aligned} \tag{31}$$

At this boundary, in case $i_1(x_1).i_2(x_2)...i_{z-1}(x_{z-1})$, removing the elements of $\mathbf{u}$ resolved as constants allows us to rewrite (1) as

$$\mathbf{v} = \sum_{\gamma=1}^{z} \mathbf{B}_{i_\gamma} u_{i_\gamma} + \mathbf{B}'_{i_1...i_z} \mathbf{u}'_{i_1...i_z} \tag{32}$$

$$\mathbf{B}'_{i_1...i_n} \mathbf{u}'_{i_1...i_z} = \mathbf{v} - \sum_{\gamma=1}^{z} \mathbf{B}_{i_\gamma} u_{i_\gamma} \tag{33}$$

It follows then that $\mathbf{u}'_{i_1...i_z}$ is the unique minimum 2-norm resultant for (33), else we could choose different elements for the vector $\mathbf{u}'_{i_1...i_{z-1}}(i_1(x_1).i_2(x_2)...i_{z-1}(x_{z-1}))$ corresponding to the elements of $\mathbf{u}'_{i_1...i_z}(i_1(x_1).i_2(x_2)...i_{z-1}(x_{z-1}))$ with the same or lower two norm while still resolving (1). But by definition, $\mathbf{u}'_{i_1...i_{z-1}}(i_1(x_1).i_2(x_2)...i_{z-1}(x_{z-1}))$ is a minimum two norm solution for (1), so this is impossible by the uniqueness property of the pseudo-inverse.

As such at the boundary condition, for case $i_1(x_1).i_2(x_2)...i_{z-1}(x_{z-1})$ we have:

$$\mathbf{u}'_{i_1 \dots i_z} = (\mathbf{B}'_{i_1 \dots i_z})^{\dagger}(\mathbf{v} - \sum_{\gamma=1}^{z} \mathbf{B}_{i_\gamma} u_{i_\gamma}) \tag{34}$$

Which is equal to the resolution of $\mathbf{u}'_{i_1 \dots i_z}$ in case $i_1(x_1).i_2(x_2)...i_{z-1}(x_{z-1}).i_z(\text{a})$. Since u_{i_1}, u_{i_2}, ... , u_{i_z}, and $\mathbf{u}'_{i_1 \dots i_z}$ are resolved equivalently in case $i_1(x_1).i_2(x_2)...i_{z-1}(x_{z-1})$, and case $i_1(x_1).i_2(x_2)...i_{z-1}(x_{z-1}).i_z(\text{a})$ at the switching condition, the system switches continuously between cases $i_1(x_1).i_2(x_2)...i_{z-1}(x_{z-1})$ and $i_1(x_1).i_2(x_2)...i_{z-1}(x_{z-1}).i_z(\text{a})$.

A similar method demonstrates continuity between case $i_1(x_1).i_2(x_2)...i_{z-1}(x_{z-1})$ and $i_1(x_1).i_2(x_2)...i_{z-1}(x_{z-1}).i_z(\text{b})$.

We will now demonstrate that, given the asserted constraint, that switching directly between any other cases is impossible. To do so, let us consider what would be required for such switching to actually occur. Suppose across k levels corresponding to u_{j_1} ... u_{j_k} a solution switches to another solution whose same levels now correspond to $u_{\tilde{j}_1}$ ... $u_{\tilde{j}_k}$, $u_{\tilde{j}_{\bar{k}}} \neq u_{j_{\bar{k}}}$, $\bar{k} = [1,k]$. Consider the lowest order of these switching occurrences, which correspond to variables u_{i_λ} and $u_{i_{\tilde{\lambda}}}$.

For a system to be in case $i_1(x_1).i_2(x_2)...i_{\lambda-1}(x_{\lambda-1}).i_\lambda(x_\lambda)$ the following must be satisfied:

$$|u_{i_\lambda}(i_1(x_1).i_2(x_2)...i_{\lambda-1}(x_{\lambda-1}))| > u_{i_\lambda max} \tag{35}$$
$$|u_{i_\Lambda}(i_1(x_1).i_2(x_2)...i_{\lambda-1}(x_{\lambda-1}))| \leq u_{i_\Lambda max},\ \forall \Lambda \neq \lambda \tag{36}$$

Likewise, for a system to be in case $i_1(x_1).i_2(x_2)...i_{\lambda-1}(x_{\lambda-1}).i_{\tilde{\lambda}}(x_{\tilde{\lambda}})$ the following must be satisfied:

$$|u_{i_{\tilde{\lambda}}}(i_1(x_1).i_2(x_2)...i_{\lambda-1}(x_{\lambda-1}))| > u_{i_{\tilde{\lambda}} max} \tag{37}$$
$$|u_{i_\Lambda}(i_1(x_1).i_2(x_2)...i_{\lambda-1}(x_{\lambda-1}))| \leq u_{i_\Lambda max},\ \forall \Lambda \neq \tilde{\lambda} \tag{38}$$

Thus, the only border case that exists between these two cases occurs when:

$$|u_{i_\lambda}(i_1(x_1).i_2(x_2)...i_{\lambda-1}(x_{\lambda-1}))| = u_{i_\lambda max} \tag{39}$$
$$|u_{i_{\tilde{\lambda}}}(i_1(x_1).i_2(x_2)...i_{\lambda-1}(x_{\lambda-1}))| = u_{i_{\tilde{\lambda}} max} \tag{40}$$
$$|u_{i_\Lambda}(i_1(x_1).i_2(x_2)...i_{\lambda-1}(x_{\lambda-1}))| \leq u_{i_\Lambda max},\ \forall \Lambda \neq \lambda, \tilde{\lambda} \tag{41}$$

which is in neither case $i_1(x_1).i_2(x_2)...i_{\lambda-1}(x_{\lambda-1}).i_\lambda(x_\lambda)$ nor $i_1(x_1).i_2(x_2)...i_{\lambda-1}(x_{\lambda-1}).i_{\tilde{\lambda}}(x_{\tilde{\lambda}})$. Thus it is impossible to switch directly between cases $i_1(x_1).i_2(x_2)...i_{\lambda-1}(x_{\lambda-1}).i_\lambda(x_\lambda)$ and $i_1(x_1).i_2(x_2)...i_{\lambda-1}(x_{\lambda-1}).i_{\tilde{\lambda}}(x_{\tilde{\lambda}})$, due to the continuity property of the pseudo-inverse. Due to the case conditions of the considered higher order switching case, it is also now evident that switching between these generalized two cases is impossible.

Finally we will show that, given the asserted constraints, that switching directly between a and b cases of the same level and variable is impossible. To do so, let us consider what would be required for such switching to actually occur. Suppose across k levels corresponding to u_{j_1} ... u_{j_k} a solution switches to another case where the corresponding a's and b's of these k variables invert. Consider the lowest order of these switching occurrences, which corresponds to variables u_{i_λ}.

For a system to be in case $i_1(x_1).i_2(x_2)...i_{\lambda-1}(x_{\lambda-1}).i_\lambda(\text{a})$ the following must be satisfied:

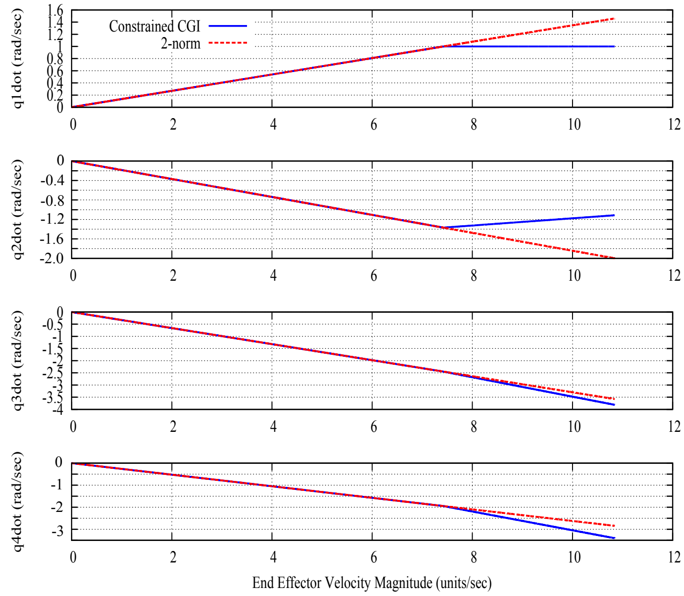

Fig. 3. Constrained CGI resolved joint velocities for increased end effector velocity magnitude, in configuration A

$$u_{i_\lambda}(i_1(x_1).i_2(x_2)...i_{\lambda-1}(x_{\lambda-1})) > u_{i_\lambda max} \tag{42}$$

Likewise, for a system to be in case $i_1(x_1).i_2(x_2)...i_{\lambda-1}(x_{\lambda-1}).i_\lambda(\text{b})$ the following must be satisfied:

$$u_{i_\lambda}(i_1(x_1).i_2(x_2)...i_{\lambda-1}(x_{\lambda-1})) < -u_{i_\lambda max} \tag{43}$$

No boundary points exist between these two cases, so by the continuity property of the pseudo-inverse, switching directly between these two cases is impossible. Due to the case conditions of the considered higher order switching case, it is also now evident that switching between these generalized two cases is impossible.

A similar method proves that direct case switching combining switches in level variables and switching of a and b cases of the same level variable are impossible, given the asserted constraints.

■

VI. IMPLEMENTATION

Fig. 3 demonstrates constrained CGI being utilized in resolving the same system used to illustrate the discontinuity in standard CGI, in section III. It is seen that the system follows standard CGI until 2-norm saturates the second variable, $\dot{q}_2$, at which point constrained CGI stops resolving the system. Standard CGI, on the other hand, continues resolving the system by discontinuously redistributing the load among $\dot{q}_2$, $\dot{q}_3$, and $\dot{q}_4$. If constrained CGI is being utilized to maintain course along the desired trajectory, other methods should be utilized such as scaling the velocities to within an attainable range.

As a result of restricting the domain it is also seen that the attainable velocity space is limited using constrained versus

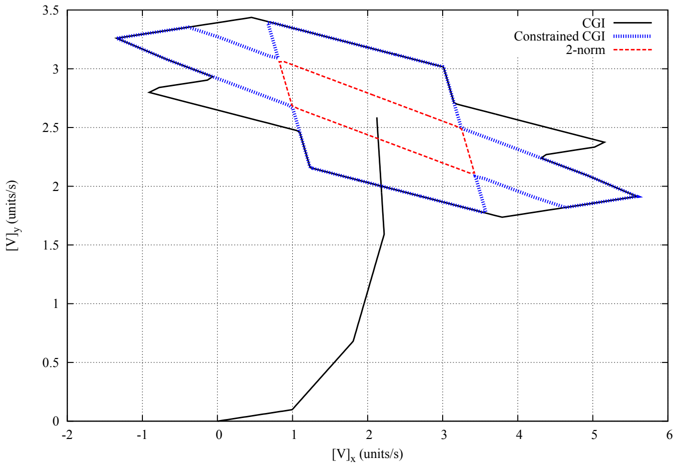

Fig. 4. Maximum achievable end effector velocities in configuration A using 2-norm, CGI, and constrained CGI resolution

unconstrained CGI. Fig. 4 illustrates the maximum achievable velocities (scaled down by a factor of 10) in all directions in configuration A, using 2-norm, CGI, and constrained CGI. It is seen that while always greater than or equal to that achieved with 2-norm, the maximum velocity of constrained CGI is always less than or equal to unconstrained CGI. As such, for applications in which the effects of discontinuities in resolution are not an issue, it would be preferable to still use unconstrained CGI.

VII. Conclusions

In the resolution of redundant systems, it often occurs that, for a desired realizable output, a resolved solution exceeds input bounds, rendering a resolution unfeasible. The Cascaded Generalized Inverse (CGI) was introduced to take these inputs limits into account and extend — more so than any other method — the resolvable output range of the popular pseudo-inverse approach to resolution.

This paper analyzes CGI and draws attention to the discontinuities which in resolution may introduce error, instability, and other undesired effects (e.g. joint knocking) in physical systems. In order to overcome these discontinuities, a constrained CGI is proposed restricting the domain of CGI — sacrificing some extended output range to ensure continuity. A numerical example demonstrates constrained CGI resolving redundancy in kinematically redundant manipulators and eliminating the discontinuities observed in CGI resolution. Proof of continuity of constrained CGI is also provided for general LTI systems.

From a wider perspective, taking into account the available redundancy resolution methods offering greater output realization than the pseudo-inverse (infinity-norm, neural-networks, CGI, etc.), the proposed constrained CGI is the only resolution method which simultaneously ensures continuity, ease of implementation, and applicability in general (low and high order) systems.

Future works include experimental implementation and verification of constrained CGI in reducing negative effects of discontinuity, such as tracking error and vibration.

References

[1] O. Khatib, “Dynamic control of manipulators in operational space,” in *6th IFToMM Congr. on Theory of Machines and Mechanisms*, New Delhi, 1983, pp. 1123-1131.

[2] V. Salvucci, Y. Kimura, O. Sehoon, and Y. Hori, “Force maximization of bi-articularly actuated manipulators using infinity norm,” *IEEE Trans. Mechatron*, vol. 18, no. 3, pp. 1080-1089, Jun. 2013.

[3] I-C. Shim and Y-S. Yoon, “Stabilized minimum infinity-norm torque solution for redundant manipulators,” *Robotica*, vol. 16, no. 2, pp. 193-205, Mar. 1998.

[4] Y. Zhang, J. Wang, and Y. S. Xia, “A dual neural network for redundancy resolution of kinematically redundant manipulators subject to joint limits and joint velocity limits,” *IEEE Trans. Neural Netw.*, vol.14, no. 3, pp. 658-667, May 2003.

[5] H. Ding and J. Wang, “Recurrent neural networks for minimum infinity-norm kinematic control of redundant manipulators,” *IEEE Trans. Syst. Man Cybern. A., Syst. Humans*, vol. 29, no. 3, pp. 269-276, May 1999.

[6] V. Salvucci, Y. Kimura, S. Oh, and Y. Hori, “Non-Linear phase different control for precise output force of biarticularly actuated manipulators,” *Advanced Robotics*, Volume 27, Issue 2, pp. 109-120, Dec. 2012.

[7] V. Salvucci, Y. Kimura, S. Oh, T. Koseki, and Y. Hori, Comparing approaches for actuator redundancy resolution in biarticularly-actuated robot arms, *IEEE/ASME Transactions on Mechatronics*, 2013.

[8] J.A.M Petersen and M. Bodson, “Constrained quadratic programming techniques for control allocation,” *IEEE Trans. Control Syst. Technol.*, vol.14, no.1, pp.91-98, Jan. 2006.

[9] J. C. Virnig and David S. Bodden. “Multivariable control allocation and control law conditioning when control effectors limit (STOVL aircraft),” in *AIAA Guidance, Navigation, and Control Conf.*, Scottsdale, AZ., 1994, pp. 572-582.

[10] K. A. Bordignon, “Constrained control allocation for systems with redundant control effectors,” Ph.D. Dissertation, Aero. Eng., Virginia Poly. Inst. and State. Univ., Blacksburg, 1996.

[11] D. Enns, “Control allocation approaches,” in *Guidance, Navigation, and Control Conf. and Exhibit*, Minneapolis, MN., 1998, pp. 98-108.

[12] R.E Beck.“Application of control allocation methods to linear systems with four or more objectives,” Ph. D. Dissertation, Aero. Eng., Virginia Poly. Inst. and State. Univ., Blacksburg, 2002.

[13] Y. Zhang, C. A. Rabbath, and C. Y. Su, “Reconfigurable control allocation applied to an aircraft benchmark model,” in *American Control Conf.*, Seattle, WA., 2008, pp. 1052-1057.

[14] Y. Zhang, V.S. Suresh, B. Jiang, and D. Theilliol, “Reconfigurable control allocation against aircraft control effector failures,” in *IEEE Int. Conf. on Control Applications*, Singapore, 2007, pp. 1997-1202.

[15] A. Marks, J.F. Whidborne, and I. Yamamoto, “Control allocation for fault tolerant control of a VTOL octorotor,” in *UKACC Int. Conf. on Control*, Cardiff, Wales, 2012, pp. 357-362.

[16] V. P. Bui, H. Kawai, Y. B. Kim, and K. S. Leei, “A ship berthing system design with four tug boats,” *J. of Mechanical Science and Technology*, vol. 25, no. 5, pp. 1257-1264. May 2011.

[17] X. Shi, Y. Wei, J. Ning, and M. Fu, “Constrained control allocation using Cascading Generalized Inverse for dynamic positioning of ships,” in *Int. Conf. on Mechatronics and Automation*, Beijing, China, 2008, pp. 1636-1640.

[18] A. Maciejewski, “Obstacle avoidance for kinematically redundant manipulators in dynamically varying environments,” *Int. J. of Robotics Research*, vol. 4, no. 3 pp. 109-117, Sep. 1985.

[19] S. Chiaverini , “Singularity-Robust task-priority redundancy resolution for real- time kinematic control of robot manipulators,” *IEEE Trans. Robot. Autom.*, Vol. 13, No. 3, pp. 398 - 410, Jun. 1997.

[20] A. Sabanovic, K. Jezernik, and K. Wada, “Chattering-free sliding modes in robotic manipulators control,” *Robotica*, vol. 14, no. 1, pp. 17-29, 1995.

[21] W. Ren, J-S. Sun, R. Beard, and T. McLain, “Nonlinear tracking control for nonholonomic mobile robots with input constraints: an experimental study,” in *Proc. American Control Conf.*, Portland, Or., 2005, pp. 4923-4928.